

[INICIAR SESIÓN](#)[NUESTROS PLANES](#)[TODOS LOS CURSOS](#)[FORMACIONES](#)[CURSOS](#)[PARA EMPRESAS](#)[ARTÍCULOS DE TECNOLOGÍA > DEVOPS](#)

# Creando volúmenes con Docker



Yuri Matheus

15/08/2022



Esse artigo faz parte da Formação DevOps



Cuando se elimina un container, se pierde toda su información. ¿Hay alguna forma de conservar la información del contenedor? Este medio se llama volumen, veamos cómo

crearlos.

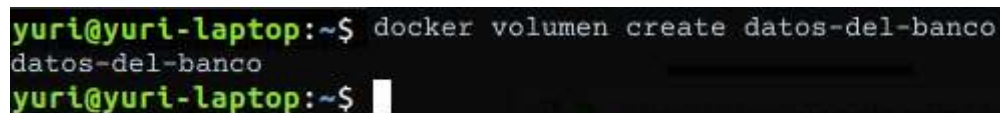
## Creando nuestro volumen

Queremos crear una copia de los datos en el **container** para nuestra máquina. Si el **container** se cae o es removido, podemos decirle dónde están los datos. De esta forma, nuestra información se guarda independientemente del estado del **container**.

Es decir, queremos decirle a **Docker** que cree un repositorio de datos para **containers** o **volumen**.

Digamos a docker que queremos crear un **volumen** con `volumen create` llamado `datos-del-banco`, en mi caso:

```
docker volumen create datos-del-banco
```



```
yuri@yuri-laptop:~$ docker volumen create datos-del-banco
datos-del-banco
yuri@yuri-laptop:~$
```

Aparentemente todo salió bien, pero ¿cómo podemos saber cuáles son nuestros volúmenes? Bueno, podemos decirle a docker que los enumere con `(ls)`

```
docker volume ls
```



```
yuri@yuri-laptop:~$ docker volumen ls
DRIVER          volumen Name
local           datos-del banco
yuri@yuri-laptop:~$
```

En este comando, Docker nos muestra el nombre del volumen y su driver. Es decir, de qué manera debe montar el volumen. En mi caso, este es el driver `local`, el driver Docker predeterminado (`built-in`).

Bien, ya tenemos un volumen creado, pero ¿cómo podemos asignarlo a un container?

## Hacer referencia a un volumen

Ya tenemos nuestro volumen creado, así que digámosle a docker que ejecute un container `container run`, llamado `db` (`--name db`), en mi caso, con nuestro volumen (`-v`) `datos-del-banco` asociado con el directorio de docker. Información que queremos guardar, `/var/lib/mysql`, en este caso:

```
docker container run --name db -v datos-del-banco:/var/lib/mysql
```

De esta forma, le estamos diciendo a Docker crear un container y asociar el directorio `/var/lib/mysql` con el volumen `datos-del-banco`. Nuestra aplicación necesita una contraseña para iniciar sesión en la **base de datos**, así que digamos que este container, en su **entorno** (`-e`, environment), tendrá la contraseña `alura`:

```
docker container run --name db -v datos-del-banco:/var/lib/mysql -e MYSQL_ROOT
```

Genial, ahora solo tenemos que hablar sobre la imagen que creará nuestro container, en nuestro caso, es la imagen de **MySQL**:

```
docker container run --name db -v datos-del-banco:/var/lib/mysql -e MYSQL_ROOT
```

```
yuri@yuri-laptop:~$ docker container run --name db datos-del-banco:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=alura mysql
Initializing database
2018-04-02T21:58:06.427757Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_for_timestamp server option (see documentation for more details).
2018-04-02T21:58:08.286718Z 0 [Warning] InnoDB: New log files created, LSN=45790
2018-04-02T21:58:08.745846Z 0 [Warning] InnoDB: Creating foreign key constraint system tables.
```

Ahora bien, nuestro container se está ejecutando, hagamos algunas pruebas para ver como funciona nuestro volumen. En otra pestaña en la terminal, digamos a docker que queremos ejecutar algunos comandos en un container (`container exec`) con una terminal interactiva (`-ti`) en nuestro container `db` con el interpretador `/bin/bash`:

```
docker container exec -it db /bin/bash
```

```
yuri@yuri-laptop:~$ docker container exec -it db /bin/bash
root@019bb53f34da:/#
```

Iniciemos sesión en **mysql** con el usuario (`-u`) `root` y la contraseña `-p alura`:

```

root@019bb53f34da:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.20 MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>

```

Vamos a crear (create) una base de datos (database) para hacer nuestra prueba, en mi caso, llamaré a esta base de datos de la tienda:

```

mysql> create database tienda
Query OK, 1 row affected (0.00 sec)

mysql>

```

Salgamos de nuestro container, para eso podemos presionar las teclas CTRL + D hasta regresar a nuestra terminal, o usar el atajo **Ctrl + P + Q**. Le decimos a docker que detenga nuestro container (container stop)db y luego le indicamos remover (container rm):

```
docker container stop db
```

```
docker container rm db
```

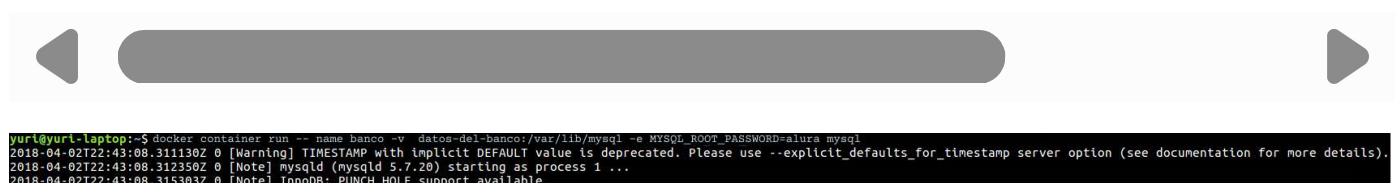
```

yuri@yuri-laptop:~$ docker container stop db
db
yuri@yuri-laptop:~$ docker container rm db
db
yuri@yuri-laptop:~$

```

Incluso habiendo retirado el container, su información debe haber sido guardada en el volumen. Vamos a crear un nuevo container, esta vez lo llamaré banco y haré referencia al mismo volumen datos-del-banco en el directorio /var/lib/mysql:

```
docker container run --name banco -v datos-del-banco:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=alura mysql
```



```

yuri@yuri-laptop:~$ docker container run --name banco -v datos-del-banco:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=alura mysql
2018-04-02T22:43:08.311130Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_for_timestamp server option (see documentation for more details).
2018-04-02T22:43:08.312350Z 0 [Note] mysqld (mysqld 5.7.20) starting as process 1 ...
2018-04-02T22:43:08.315303Z 0 [Note] InnoDB: PUNCH HOLE support available

```



Veamos si nuestra información fue guardada, accedamos a este nuevo container con el comando `docker container exec -it banco /bin/bash` y accedamos a `mysql`

```
docker container exec -it banco /bin/bash
```

```
mysql -u root -p
```

```
yuri@yuri-laptop:~$ docker container exec -it banco /bin/bash
root@f36c4f5604bb:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.20 MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Digamos a docker que muestre (**show**) las bases de datos existente (**database**):

```
show databases;
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| tienda |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql>
```

Observa que nuestra base de datos de la tienda aparece normalmente, es decir, pudimos mantener los datos incluso después de que se eliminó el contenedor

## Para saber más

Podemos referenciar más de un container para el mismo volumen. Es decir, podemos tener dos o más containers haciendo referencia al mismo volumen.

Estos volúmenes se almacenan en el directorio `/var/lib/docker/volumen/`, sin embargo, también podemos crear volúmenes en otros directorios. Basta que al momento de la creación del container, pasemos como parámetro el camino del directorio en lugar del nombre del volumen

```
docker container run --name db -v /outro/diretorio:/var/lib/mysql -e MYSQL_RO
```

Hemos visto cómo crear un volumen, pero ¿cómo podemos eliminarlo? Podemos decirle a docker que elimine el volumen (`volumen rm`):

```
docker volumen rm datos-del-banco
```

```
yuri@yuri-laptop:~$ docker volumen rm datos-del-banco
datos-do-banco
yuri@yuri-laptop:~$
```

Recordando que para remover un volumen, ningún container puede estar usándolo.

Docker ha ganado mucho espacio tanto en el escenario de desarrollo como de infraestructura, porque viene con el objetivo de facilitar el desarrollo de aplicaciones.

Aquí en Alura Latam contamos con un [Curso de Docker](#) en el cual aprenderás todo sobre qué es un container, cómo hacer que se comuniquen, además de aprender a crear tus propias **imágenes** para personalizar tus containers.



Yuri Matheus

Yuri es desarrollador e instructor. Es estudiante de Sistemas de Información en la FIAP y se graduó como Técnico en Computación en el Senac SP. Su enfoque está en las plataformas Java y Python y otras áreas como la arquitectura de software y el aprendizaje automático. Yuri también trabaja como editor de contenido en el blog de Alura, donde escribe principalmente sobre Redes, Docker, Linux, Java y Python.

Este artículo fue adecuado para Alura Latam por: [Jose Charris](#)



Esse artigo faz parte da  
Formação DevOps

Cursos de DevOps

ARTÍCULOS DE TECNOLOGÍA > DEVOPS

**En Alura encontrarás variados cursos sobre DevOps. ¡Comienza ahora!**

**SEMESTRAL**

**US\$49,90**

un solo pago de US\$49,90

- ✓ 218 cursos
- ✓ Videos y actividades 100% en Español
- ✓ Certificado de participación
- ✓ Estudia las 24 horas, los 7 días de la semana
- ✓ Foro y comunidad exclusiva para resolver tus dudas

- ✓ Acceso a todo el contenido de la plataforma por 6 meses

**¡QUIERO EMPEZAR A ESTUDIAR!**

[Paga en moneda local en los siguientes países](#)

**ANUAL**

**US\$79,90**

un solo pago de US\$79,90

- ✓ 218 cursos
- ✓ Videos y actividades 100% en Español
- ✓ Certificado de participación
- ✓ Estudia las 24 horas, los 7 días de la semana
- ✓ Foro y comunidad exclusiva para resolver tus dudas
- ✓ Acceso a todo el contenido de la plataforma por 12 meses



**¡QUIERO EMPEZAR A ESTUDIAR!**

[Paga en moneda local en los siguientes países](#)

Acceso a todos  
los cursos

Estudia las 24 horas,  
dónde y cuándo quieras

Nuevos cursos  
cada semana

## NAVEGACIÓN

PLANES  
INSTRUCTORES  
BLOG  
POLÍTICA DE PRIVACIDAD  
TÉRMINOS DE USO  
SOBRE NOSOTROS  
PREGUNTAS FRECUENTES

## ¡CONTÁCTANOS!

¡QUIERO ENTRAR EN CONTACTO!

## BLOG

PROGRAMACIÓN  
FRONT END  
DATA SCIENCE  
INNOVACIÓN Y GESTIÓN  
DEVOPS

AOVS Sistemas de Informática S.A  
CNPJ 05.555.382/0001-33

## SÍGUENOS EN NUESTRAS REDES SOCIALES



## ALIADOS



En Alura somos unas de las Scale-Ups seleccionadas por Endeavor, programa de aceleración de las empresas que más crecen en el país.



Fuimos unas de las 7 startups seleccionadas por Google For Startups en participar del programa Growth Academy en 2021

POWERED BY

## CURSOS

### Cursos de Programación

Lógica de Programación | Java

### Cursos de Front End

HTML y CSS | JavaScript | React

### Cursos de Data Science

Data Science | Machine Learning | Excel | Base de Datos | Data Visualization | Estadística

### Cursos de DevOps

Docker | Linux

### Cursos de Innovación y Gestión

Productividad y Calidad de Vida | Transformación Ágil | Marketing Analytics | Liderazgo y Gestión de Equipos | Startups y Emprendimiento